Trino for lakehouses, data oceans, and beyond

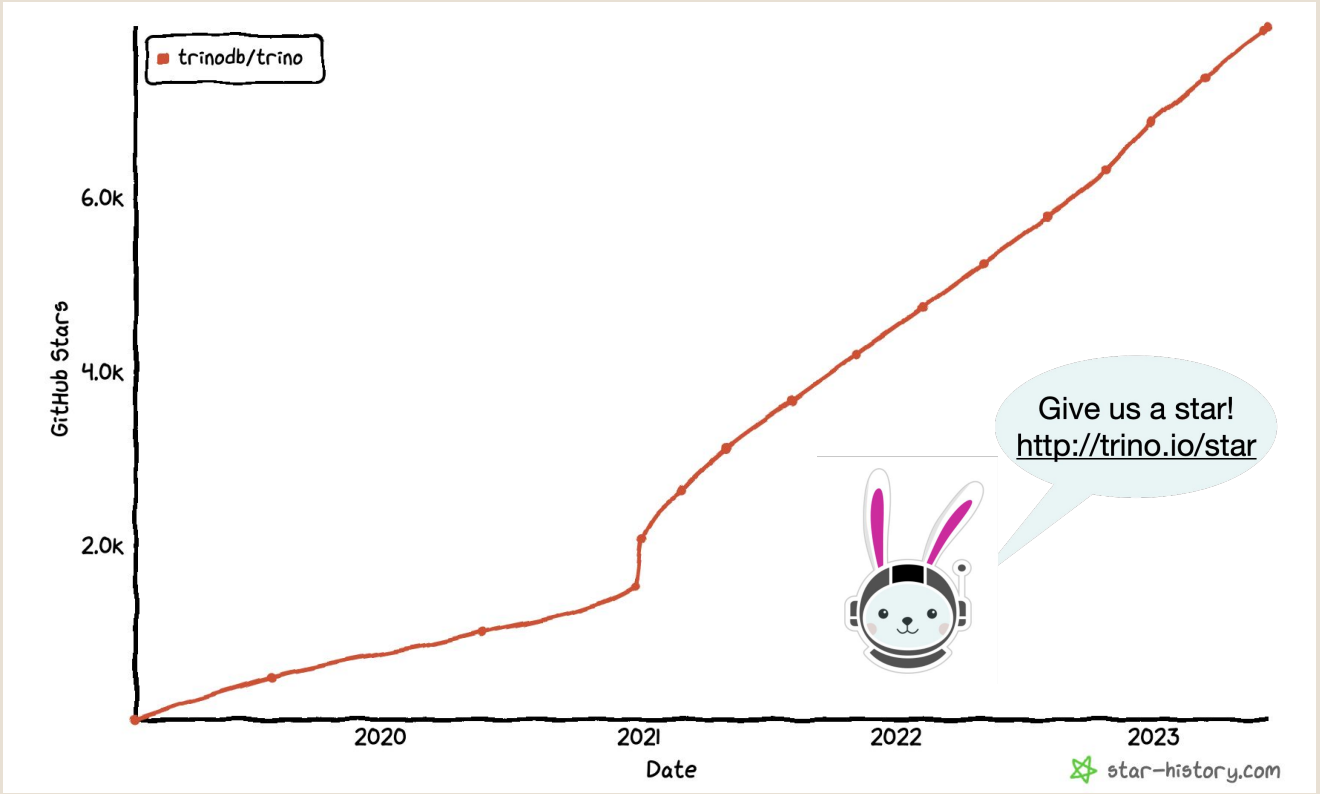
A project update for June 2023

Martin Traverso

By the numbers

- **16** releases since Trino Summit in Nov 2022
- **2,250** commits in 2023 so far (32,505 total)
- **660+** contributors
- **9,900+** Slack members
  - Go join and be #10,000!
- Over **20,000** known members in the community
  - From over **1,800** companies
- Continued rise on db-engines ranking
  - Climbed from **96th to 69th** in one year

Continued growth > 8k



New maintainers

- James Petty
  - Amazon Athena and EMR team
  - Deep and very wide experience as developer on Trino

- Manfred Moser
  - Starburst and Trino developer relations team
  - Trino: The Definitive Guide and documentation author
  - Trino Community Broadcast host

- https://trino.io/development/roles.html

What's happened since Trino Summit 2022?

Table function improvements

- **`exclude_columns`** to simplify access for wide tables

```
SELECT *
FROM TABLE(exclude_columns(
                input => TABLE(person),
                columns => DESCRIPTOR(dob, ssn)))
```

- **`sequence`** to generate number sequence

```
SELECT *
FROM TABLE(sequence(
                start => 1,
                stop => 1000,
                step => 10))
```

Table function improvements

- Pass-through with **`query`**/**`raw_query`**
  for more connectors, for example Elasticsearch

```
SELECT
  *
FROM
  TABLE(
    example.system.raw_query(
      schema => 'sales',
      index => 'orders',
      query => '{
        "query": {
          "match": {
            "name": "ALGERIA"
          }
        }
      }'
    )
  );
```

- **`procedure`** in SQL Server connector
  for stored procedure invocation

```
SELECT *
FROM TABLE(example.system.procedure(
            query => 'EXECUTE people.employee_sp'));
```

Fault-tolerant execution

- Simplified configuration
- Improved performance
- Support for exchange spooling on HDFS
- Support for MongoDB, BigQuery, Redshift, and Oracle connector

Ideal for your lakehouse migration and other ETL workloads!

Schema evolution, (meta)data, and tools

Enabling advanced lakehouse and related use cases:

- **`ALTER COLUMN … SET DATA TYPE`**
- **`ALTER TABLE … RENAME COLUMN`** added to connectors
- Nested fields in **`DROP COLUMN`**
- **`$timeline`** system table in Hudi
- **`table_changes`** function and views in Delta Lake
- Support for REST, JDBC, and Nessie catalogs for metadata in Iceberg
- Lots of metadata tables in Iceberg

Lakehouse migration - table procedures

- **`migrate`** to go from Hive to Iceberg

```
CALL example.system.migrate(
    schema_name => 'testdb',
    table_name => 'customer_orders')
```

- **`register_table / unregister_table`**

```
CALL example.system.register_table(
    schema_name => 'testdb',
    table_name => 'customer_orders',
    table_location => 'hdfs://lake:9000/user/hive/warehouse/customer_orders-58
```

Tons of performance improvements

- Replaced Hive and Hadoop dependencies.
- Lower resource utilization when inserting into tables.
- Faster **`INSERT`** queries with fault-tolerant execution.
- Faster query planning with many **`GROUP BY`** clauses.
- Faster queries on large clusters with skewed queries.
- Faster comparison between date columns and timestamps.
- Faster parsing of date values.
- Faster queries with selective joins.
- Faster queries when predicates are null.
- Faster queries with joins on aggregations.
- Faster aggregations with **`DISTINCT`** clauses.
- Faster **`LIKE`** expressions with dynamic patterns.
- Faster queries with predicates with the **`year`** function.
- Reduced memory usage by the coordinator.
- Improved performance of table functions with table args.
- Lower latency for small queries with fault-tolerant execution.
- Faster queries with window operations or pattern recognition on small partitions.
- Faster **`row_number`** and **`rank`** functions.
- Faster results when queries return no values.
- Faster aggregations with variable file sizes.
- Faster **`INSERT`**, **`CREATE TABLE AS … SELECT`**, and **`EXECUTE`** when table stats are wrong or missing.
- Faster queries with **`LIKE`** expressions that contain a **`%`**.
- Faster **`UNION ALL`** queries.
- Faster query planning for queries with joins.
- Faster **`sum`** aggregations.
- Faster **`filter`** function on arrays.

In case that wasn’t enough…

Lakehouse Connectors

- Faster joins on partition columns.
- Faster reading and writing of Parquet files.
- Reduced memory usage when reading/writing Parquet files.
- Faster queries that read a small number of columns.
- Faster **`DROP TABLE`**.
- Faster queries with with filters or projections on low-cardinality string columns.
- Improved performance on tables created by Trino.
- Improved performance on tables written to by Trino.
- Improved performance when table statistics aren’t available.
- Faster joins when both sides of a join have keys with the same definition.
- Faster planning for **`SELECT`** queries.

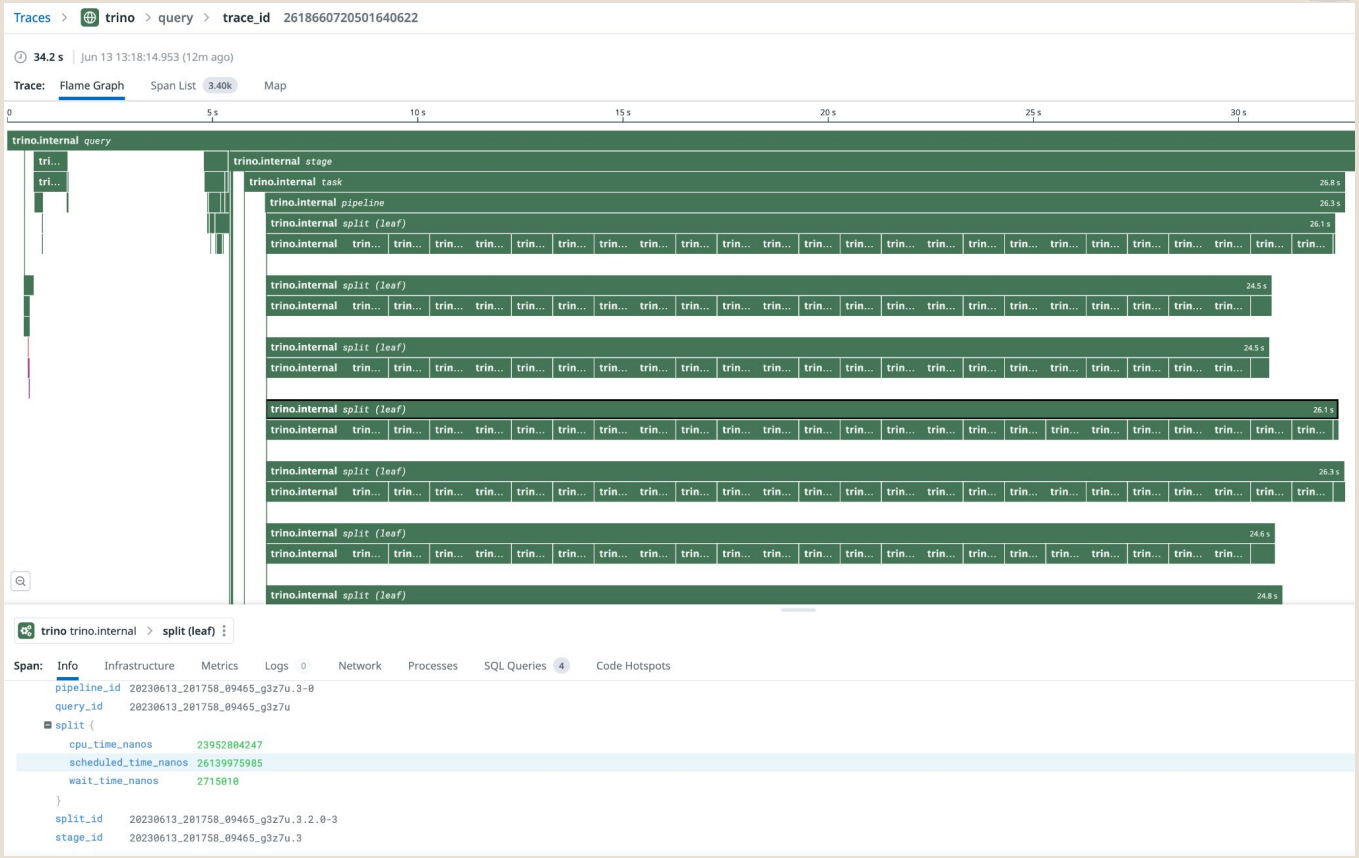
Lakehouse Connectors, cont.

- Faster planning for tables with large numbers of columns.
- Faster scans for **`COUNT(*)`** on row-oriented formats.
- Faster queries with selective filters on `row` columns.
- Faster joins on partition columns.

Other Connectors

- Support for Apache Arrow when reading from BigQuery.
- Faster queries when latency between Trino and the server is high in Oracle, PostgreSQL, and Redshift.
- Faster queries when selecting a small number of columns in Oracle, PostgreSQL, and Redshift.
- Faster queries with **`sum(DISTINCT …)`** in JDBC connectors.

Tracing with OpenTelemetry



Client tool news

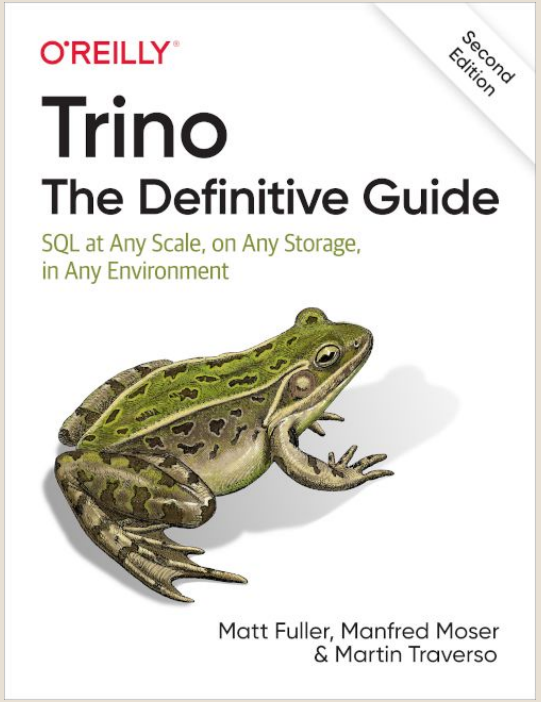
- Heavy investment into the Python client
  - Various DB-API support improvements
  - Updated to support SQLAlchemy 2.0
  - **`EXECUTE IMMEDIATE`** and other performance improvements
  - Variable precision date/time and other improved type support
- dbt Cloud support available
- More to come from the Python ecosystem later today and tomorrow

Roadmap

- SQL 2023
  - JSON enhancements
  - Numeric literal usability enhancements
- `json_table` function
- Snowflake connector
- Java 21 soon
- Project Hummingbird

Trino: The Definitive Guide

- Second edition available
- Get a digital, English version [for free from Starburst]
- Polish edition now available
- Chinese edition coming soon



Starburst

Getting involved

- Join community chat on Slack https://trino.io/slack.html
- Contribute code
  - https://trino.io/development
  - “**good first issue**” tag on github
- File issues and bugs at https://github.com/trinodb/trino
- Write blog posts and join us as guest on Trino Community Broadcast

Questions?

Hudi

- `$timeline` system table for tracking a table's timeline
- Improved performance from parallelized split generation and processing

Delta Lake

- Support for views
- Support for writing to the change data feed in Delta Lake
- Support for Databricks 12.2 LTS
- `unregister_table` procedure
- `table_changes` function

Iceberg

- Support REST, JDBC, and Nessie catalogs for metadata
- `unregister_table` procedure
- Support for sorted tables
- Better file pruning

The Iceberg connector work is making Trino a first-class query engine for Iceberg users.

Hive

- `migrate` procedure to convert a Hive table to Iceberg
- Faster join queries over Hive bucketed tables.
- Faster planning for tables with many columns in Hive.

Slack updates

- Getting close to 10k users with average of 1000 active users per day
- Language channels to enable more users
    - general-de , general-ko, general-cn, general-jp
- Vendor channels to enable community partners and users
    - vendor-starburst, vendor-aws
- Ecosystem, tool, and technology channels as requested

Presentations and meetings

Help us spread the word about Trino!

- Ready made for you to use, and contribute more
  - https://github.com/trinodb/presentations
  - https://trinodb.github.io/presentations/
- Events calendar for your meetings and talks
- Support for community meetings such as Starburst 101 and Trino on Ice

Releases 405-420 highlights

[Trino 405]({{site.url}}/docs/current/release/release-405.html)

* Support for `ALTER COLUMN ... SET DATA TYPE` statement.
* Support for Apache Arrow when reading from BigQuery.
* Support for views in the Delta Lake connector.
* Support for the Iceberg REST catalog.
* Support for Protobuf encoding in the Kafka connector.
* Support for fault-tolerant execution in the MongoDB connector.
* Support for `DELETE` and query pushdown in the Redshift connector.
* Performance improvements when reading Parquet data.

[Trino 406]({{site.url}}/docs/current/release/release-406.html)

* Support for JDBC catalog in the Iceberg connector.
* Support for fault-tolerant execution in the BigQuery connector.
* Support for exchange spooling on HDFS.
* Support for `CHECK` constraints with `INSERT` statements.
* Improved performance for Parquet files with the Delta Lake, Hive, Hudi and Iceberg connectors.

Performance improvements

Incremental improvements to connector performance and core performance are part of every release. A few highlights:

- Faster aggregations containing `DISTINCT`
- Faster `LIKE` with dynamic patterns
- Improved performance of queries involving window functions or `MATCH_RECOGNIZE`
- Faster `UNION ALL` queries
- Faster `sum(`DISTINCT ...`)` queries for various connectors

Lakehouse performance improvements

- Replaced Hive and Hadoop dependencies.
- Faster Parquet data processing.
  - *Many* incremental performance improvements.
- Faster reads of nested row fields in Delta Lake connector.
- Faster joins on partition columns in lakehouse connectors.